COL774 ASSIGNMENT 2

Aryan Jain
2019CS10334

Q1. Naive Bayes

* We implement a naive bayes classifier which predicts the rating given to a review.

Part a)

* We implement Naive Bayes on the reviewText without any preprocessing on the data.

* Rating can take values from [1, 2, 3, 4, 5] and hence the classifier required is a 5-way classifier.

* We use Laplace Smoothing with a smoothing coefficient of alpha = 1 to avoid any zero probabilities for words that didn't occur in the training set for the corresponding rating.

* We implement the model using logarithms to avoid overflows when predicting the rating for a review

* Accuracy on the training data is : 72.08%
* Accuracy on the test data is : 66.54%

Part b)

* If we randomly predict the rating of the reviews in the test data we get an accuracy of : 16.574%

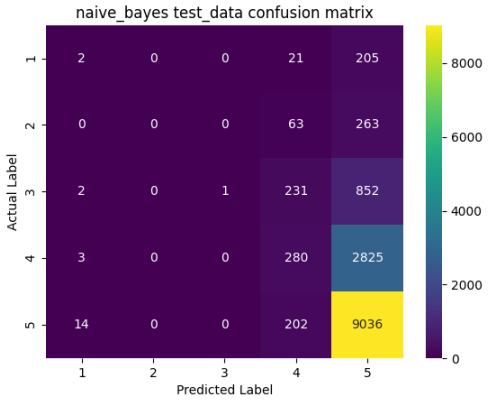
* If we always predict the most occurring rating for all reviews in the test data (a rating of 5) we get an accuracy of : 51.864%

* The Naive Bayes model trained on unprocessed data is 4.01 times better than randomly predicting ratings and 1.3 times better than always predicting the most occuring rating.

Part c)

* The confusion matrix helps us understand which classes are incorrectly predicted by our classifier and as which ones.

* The confusion matrix for the Naive Bayes classifier trained on raw data is :



* The rating = 5 has the largest diagonal entry which means a review with a rating of 5 is most likely to be correctly classified.

* The classifier almost never predicts 2 or 3.

* We can also observe that an extremely imbalanced distribution of classes in the data leads to a skewed model, especially

when the data is very similar across the data set.

Part d)

* We use the NLTK library to filter out stop words and stem the remaining words.

* We first use the English stop-words corpus from the NLTK library and ignore any words in the text present in the stop_words set.

* Next we stem each word before adding it to the dictionary, or incrementing it's count if already present.

* The test accuracy obtained for the new trained model with stop words removed and stemming is : 61.25%

* There is a decrease in accuracy which indicates that stemming removed some uniqueness from the reviews which was derived from the different usage of the different forms of the same words.

Part e)

* Feature 1 : Bi - Grams

* We pair up the words in the text to form bi grams which are then used as terms in our dictionary

* The accuracy on the train set with bi-grams is : 77.68 %

* The accuracy on the test set on using bi-grams is : 66.14 %

* Feature 2 : Tri - Grams

* We group up word triplets in the text to form tri-grams which are then used as terms in our dictionary

* The accuracy on the train set with tri-grams is : 88.65 %

* The accuracy on the test set on using bi-grams is : 66.19 %

* The increase in the accuracy for the train set for the above 2 features shows that grouping up consecutive words helps in increasing the uniqueness of the terms corresponding to each rating class.

* The lack of effect on the test set shows that the terms, even though unique to the rating class, are not well represented in the test data set and hence are not helpful in classifying the test data.

* Feature - 3 : Removing words with less frequency of occurrence

* We remove any word which occurs less than 50 times in the entire training data set.

* The accuracy on the train set with is : 51.32 %

* The accuracy on the test set is : 66.19 %

* The decrease in accuracy is justified as this feature eradicates any words that are uniquely present in reviews of a certain rating class.

Part f)

* The F1-score is a measure of a model's accuracy on a certain data set. For any particular class, it is calculated as the harmonic mean of the precision and the recall.

* Precision is the fraction of true-positives over the total number of examples.
Recall is the fraction of true-positives over the total number of positives.

Class 1 : 0.0002811357885885887
Class 2 : 0
Class 3 : 0.0001325732467188121 3
Class 4 : 0.03273322422258593
Class 5 : 0.7772234646482022

* The F-score for Class 2 is 0 because there are no true-positives for class 2.

* The macro-F1 score is the arithmetic mean of the F1 scores for the individual classes.

macro-F1 = 0.16207407958121858

* The F1-score is a more suited measure for this kind of a classifier as it gives us a lot more insight into the kind of predictions made by the classifier. From the above F1-scores we can tell that the classifier is making a ridiculously large amount of class 5 predictions which is leading to a moderate accuracy only because the rating 5 is abundant. And it is barely ever making a correct prediction for any other class.

Part g)

* Method 1 : Words in the reviewText that are also present in the summary are counted 10 times

* The accuracy on the train set is : 75.55%
* The accuracy on the test set is : 66.87%

* Method 2 : Adding the summary 10 times at the end of the reviewText

* The accuracy on the train set is : 90.14%
* The accuracy on the test set is : 67.15%

* The above results show that words present in the summary are significantly unique to the corresponding rating class, and hence we see an improvement in the accuracy.

Q2 MNIST Digit Classification

* We use SVMs with a linear or gaussian kernel to build a handwritten digit classifier.

* We use the library CVXOPT which is used to solve convex optimization problems optimally.

2 a) Binary Classification

* D = 4 and (D+1) mod 10 = 5
* We take the subset of the images that are either in class 4 or 5 of both the train data and the train data. We then use the train data to train a SVM and test it using the test data.
* We take class 4 as positive and class 5 as negative when training the data or generating predictions.

Part i)

* We train a linear kernel SVM using CVXOPT.

* The accuracy on the train set is : 100%
* The accuracy on the test set is : 98.665%

* A 100% training set accuracy shows that the data in the training set is linearly separable.

Part ii)

* We transform the data to gaussian space and train the SVM using CVXOPT in this gaussian space.

* The predictions are also made by transforming all input to gaussian space, and w is also calculated while generating the predictions and is transformed to the gaussian space as well.

* The accuracy on the train set is : 100%
* The accuracy on the test set is : 99.893%

Part iii)

* We train a linear and gaussian svm using the library LIBSVM

* test accuracy of the linear SVM : 98.66%
* test accuracy of the gaussian SVM : 99.89%

* The training times are:

i) my linear svm = 53.062 s
ii) my gaussian svm = 47.32 s
iii) libsvm linear svm = 0.969 s
iv) libsvm gaussian svm = 4.65 s

* The prediction time is almost negligible, hence the libsvm models are a lot more efficient than the ones trained using CVXOPT.

2 b) Multi-Class Classification

* We train 10C2 classifiers. To make a prediction we generate prediction values for each classifier and then take the class with the highest aggregate value as the prediction.

Part i)

* Training 10C2 gaussian SVM classifiers using CVXOPT.

* The accuracy on the test set is : 97.20%

Part ii)

* Training 10C2 gaussian SVM classifiers using LIBSVM.

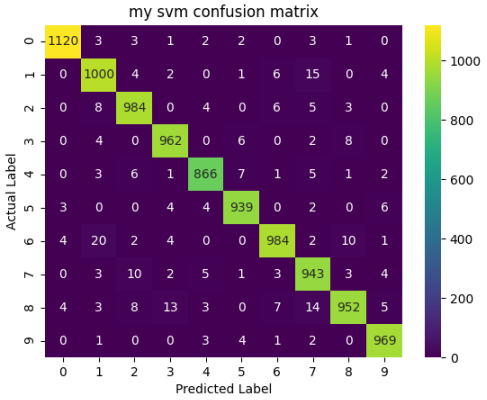
* The accuracy on the test set is : 97.25%

* The training time for my SVMs is 35 minutes

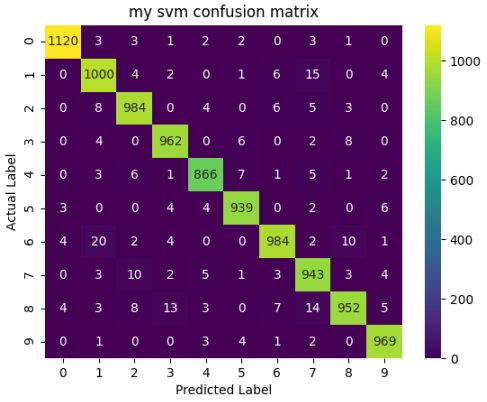
* The training time for the libsvm SVMs is 10 minutes, which makes the libsvm implementation a lot more efficient.

Part iii)

* Confusion matrix for my gaussian SVM matrices



* Confusion matrix for the LIBSVM gaussian classifiers



* Most common mispredictions:
6 mispredicted as 1
1 mispredicted as 7
8 mispredicted as 7
8 mispredicted as 3
7 mispredicted as 2
6 mispredicted as 8
2 mispredicted as 1

* Most of these mispredictions make sense, for e.g. It is very common for even humans to predict 1 as 7, 8 as 3, 3 as 8 or 6 as 8.